

REMARKS

The Office Action, mailed October 4, 2006, considered claims 1-49. Claims 1-17, 19-22, 25-31, 42, 43, 45-47 and 49 were rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Patent No.: 5,974,470 to *Hammond*. Claims 32, 39 and 41 are rejected under 35 U.S.C. 103(a) as being unpatentable over *Hammond* and further in view of U.S. Patent No.: 6,185,734 to *Saboff*.¹

By this amendment claims 1, 5, 6, 8, 9, 11-14, 16-19, 21, 24, 26, 27, 30, 32, 39, 41-43 and 46 have been amended.² Claims 1, 16, 32, 39, 41 and 42 are the only independent claims at issue.

The present invention is generally directed to allowing a software application to run using a specified version of one or more shared assemblies. For example, claim 1 defines receiving a request from an application to load an assembly, the request not including assembly version data, including when the assembly is among a plurality of assemblies located in a same directory. Next, claim 1 defines building an activation context based on a manifest, the manifest being associated with the application that requested the loading of the assembly, wherein the activation context maps global, version independent names to a particular version of an assembly, and distinguishes between versions of assemblies based on the version indicated by the manifest, the activation context being associated with the application in response to the application's request to load an assembly. Next, claim 1 defines consulting information in a manifest associated with the application, the manifest being stored separately from the application and any changes made to the manifest being implemented without having to recompile the manifest, the manifest being used to identify a particular version of the requested assembly in response to the building of the activation context. Lastly, claim 1 defines providing the particular version of the assembly for use by the application.

Claims 16 is a computer program product claim that defines building an activation context based on a manifest, the manifest being associated with an application, wherein the activation context maps global, version independent names to a particular version of an

¹ Although the prior art status of the cited art is not being challenged at this time, Applicant reserves the right to challenge the prior art status of the cited art at any appropriate time, should it arise. Accordingly, any arguments and amendments made herein should not be construed as acquiescing to any prior art status of the cited art.

² Support for the amendments to the claims are found throughout the specification and previously presented claims, including but not limited to paragraphs [0007], [0009], [0060], [0061 and Figures 2 & 9.

assembly, and distinguishes between versions of assemblies based on the version indicated by the manifest, the activation context being associated with the application in response to the application's request to load an assembly. Next, claim 16 defines interpreting the dependency information associated with the application, the dependency information identifying at least one particular version of an assembly including when the assembly is among a plurality of assemblies having at least some components located in a same directory. Lastly, claim 16 defines associating with the application at least one mapping based on the dependency information, each mapping relating a version independent assembly name that the application may provide to a version specific assembly identified in the dependency information.

Claim 32 is a data structure claim that defines a first data store operable to store a first set of data comprising a name of an assembly including when the assembly is among a plurality of assemblies located in a same directory, a second data store operable to store a second set of data comprising a version of the assembly, a third data store operable to store a third set of data comprising at least one item of the assembly, and a fourth data store operable to store a fourth set of data comprising binding path data to each item in the third set of data, wherein each data store is operable to provide information to an activation context based on a manifest, the manifest being associated with an application, wherein the activation context maps global, version independent names to a particular version of an assembly, and distinguishes between versions of assemblies based on the version indicated by the manifest when the application is executed.

Claims 39 is a data structure claim that defines a first data store operable to store a first set of data comprising a version independent name of an assembly including when the assembly is among a plurality of assemblies located in a same directory. Next, claim 39 defines a second data store operable to store a second set of data comprising a filename path to a specific version of the assembly, wherein the second set of data is associated with the first set of data such that a reference to the version independent name in the first set of data is mapped to the specific version of the assembly via the second set of data and wherein each data store is operable to provide information to an activation context based on a manifest, the manifest being associated with an application, wherein the activation context maps global, version independent names to a particular version of an assembly, and distinguishes between versions of assemblies based on the version indicated by the manifest when the application is executed.

Claim 41 is a data structure claim that defines a first data store operable to store a first set of data comprising a version independent object class name, a second data store operable to store a second set of data comprising an assembly name corresponding to a file that contains an object class that corresponds to the object class name in the first set of data including when the file is among a plurality of files located in a same directory, and a third data store operable to store a third set of data comprising a version specific name that corresponds to the first set of data such that a reference to the version independent name in the first set of data is mapped to the specific version of the object class, wherein each data store is operable to provide information to an activation context based on a manifest, the manifest being associated with an application, wherein the activation context maps global, version independent names to a particular version of an assembly, and distinguishes between versions of assemblies based on the version indicated by the manifest when the application is executed when an application is executed.

Claim 42 is a system claim that defines an initialization mechanism configured to interpret dependency data associated with an application, the dependency data corresponding to at least one assembly version on which the application depends, each assembly version corresponding to an assembly having version information associated therewith and contained in a directory structure among a plurality of assemblies. Next, claim 42 defines an activation context based on a manifest, the manifest being associated with an application, wherein the activation context maps global, version independent names to a particular version of an assembly, and distinguishes between versions of assemblies based on the version indicated by the manifest, the activation context associated with the application and constructed by the initialization mechanism based on the dependency data, the activation context relating at least one version independent assembly identifier provided by the application to a version specific assembly. Lastly, claim 42 defines a version matching mechanism configured to access the activation context to relate a version independent request from the application to a version specific assembly.

Applicants respectfully submit that the cited art of record does not anticipate or otherwise render the amended claims unpatentable for at least the reason that the cited art does not disclose, suggest, or enable each and every element of these claims.

Hammond describes a system for managing DLL modules. Specifically, *Hammond* allows for more accurate loading of the correct DLL, multiple simultaneous use of DLL's with the same name, and tracking of the run-time usage of modules (Col. 3:50-60). *Hammond* provides enhanced load logic, search logic and module version logic (Col. 4:36-38). Moreover, in *Hammond*, when a module is already being used, it assigns an alias name to the module in the load function and copies the DLL to an alias directory under an alias name. The application then uses the alias-named DLL to run the application from the alias directory (Col. 7:5-20).

Saboff describes a system and method for managing changes in software component versions and allowing multiple versions to be used simultaneously (Col. 2:34-36). *Saboff* further describes creating a centralized registry that tracks which version each application is using (Col. 2:37-40). Libraries can be updated/deleted/re-linked on-the-fly (without a reboot) using the centralized registry (Col. 3:1-7). Furthermore, *Saboff* describes having debug versions and language-specific versions of files linked by the centralized registry (Col. 4:3-4, 19-20).

Neither *Hammond* nor *Saboff* teaches or suggests building an activation context based on a manifest, the manifest being associated with the application that requested the loading of the assembly, wherein the activation context maps global, version independent names to a particular version of an assembly, and distinguishes between versions of assemblies based on the version indicated by the manifest, the activation context being associated with the application in response to the application's request to load an assembly, as recited in claim 1. Furthermore, neither *Hammond* nor *Saboff* teaches or suggests consulting information in a manifest associated with the application, the manifest being stored separately from the application and any changes made to the manifest being implemented without having to recompile the manifest, the manifest being used to identify a particular version of the requested assembly in response to the building of the activation context, as recited in claim 1. At least for either of these reasons, claim 1 patentably defines over the art of record. At least for either of these reasons, claims 16, 32, 39, 41 and 42 also patentably define over the art of record. Since each of the dependent claims depend from one of claims 1, 16, 32, 39, 41 and 42, each of the dependent claims also patentably define over the art of record for at least either of the same reasons.

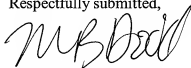
In view of the foregoing, Applicant respectfully submits that the other rejections to the claims are now moot and do not, therefore, need to be addressed individually at this time. It will be appreciated, however, that this should not be construed as Applicant acquiescing to any of the

purported teachings or assertions made in the last action regarding the cited art or the pending application, including any official notice. Instead, Applicant reserves the right to challenge any of the purported teachings or assertions made in the last action at any appropriate time in the future, should the need arise. Furthermore, to the extent that the Examiner has relied on any Official Notice, explicitly or implicitly, Applicant specifically requests that the Examiner provide references supporting the teachings officially noticed, as well as the required motivation or suggestion to combine the relied upon notice with the other art of record.

In the event that the Examiner finds remaining impediment to a prompt allowance of this application that may be clarified through a telephone interview, the Examiner is requested to contact the undersigned attorney.

Dated this 5th day of January, 2007.

Respectfully submitted,



RICK D. NYDEGGER
Registration No. 28,651
MICHAEL B. DODD
Registration No. 46,437
GREGORY R. LUNT
Registration No. 57,354
Attorneys for Applicant
Customer No. 47973